**Introduction to Relational, Logical Operators, Selection Statements**

**LAB # 03**



**Spring 2021**

**CSE102L Computer Programming Lab**

Submitted by: **Maaz Habib**

Registration No.: **20PWCSE1952**

Class Section: **C**

“On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work.”

Student Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Submitted to:

**Engr. Abdullah Hamid**

(17 May 2021)

Department of Computer Systems Engineering

University of Engineering and Technology, Peshawar

**Objectives:**

* To be familiar with Relational &amp; Logical Operators
* To understand the programming knowledge using Selection Statements (if, if-else, if-else ladder, Nested if-else)

**Task Titles:**

1. Display the largest among three numbers using if else statement?

2. Check whether a number is even or odd?

3. Check the greater of two numbers using ternary operator?

4. Write a program where you print you take a number from the user and then print your

name and registration number times the number (hint use a loop).

5. Write a program that asks a number and test the number whether it is multiple of 5 or not,

divisible by 7 but not by eleven. (all three conditions should match)

6. Check whether the entered character is vowel or consonant?

7. Write a program that takes the weekday number as input from user and print the day

name of week

E.g., Print Monday if weekday number is equal to 1. Similarly, check condition for all 7

days and print the corresponding day name. Print an error message if an invalid number is

entered.

8. Write a C++ program to enter month number between (1-12) and print number of days in

month.

9. Write a program to calculate and print the Electricity bill of a given customer. The

customer id and unit consumed by the user should be taken from the keyboard and

display the total amount to pay to the customer. The charges are as follow:

**Logical operator:**

 logical operator is a symbol used to connect two or more than expressions, such that the value of the compound expression produced depends only on that of the original expressions and on the meaning of the operator.

**Common logical**:

* AND
* OR
* NOT.

**Discussion**

In expressions that yield Boolean data type values are divided into two groups. One group uses the relational operators within their expressions and the second group uses logical operators within their expressions.

The logical operators are often used to help create a test expression that controls program flow. This type of expression is also known as a Boolean expression because they create a Boolean answer or value when evaluated. There are three common logical operators that give a Boolean value by manipulating other Boolean operand(s). Operator symbols and/or names vary with different programming languages:

The vertical dashes or piping symbol is found on the same key as the backslash \. You use the SHIFT key to get it. It is just above the Enter key on most keyboards. It may be a solid vertical line on some keyboards and show as a solid vertical line on some print fonts.

In most languages there are strict rules for forming proper logical expressions.

An example is:

6 > 4 && 2 <= 14  
6 > 4 and 2 <= 14

This expression has two relational operators and one logical operator.  Using the precedence of operator rules the two “relational comparison” operators will be done before the “logical and” operator. Thus:

true && true  
True and True

The final evaluation of the expression is:  **true.**

We can say this in English as:

It is true that six is greater than four and that two is less than or equal to fourteen.

When forming logical expressions programmers often use parentheses (even when not technically needed) to make the logic of the expression very clear.  Consider the above complex Boolean expression rewritten:

(6 > 4) && (2 <= 14)  
(6 > 4) and (2 <= 14)

Most programming languages recognize any non-zero value as true. This makes the following a valid expression:

6 > 4 && 8  
6 > 4 and 8

But remember the order of operations. In English, this is six is greater than four and eight is not zero. Thus,

true && true  
True and True

To compare 6 to both 4 and 8 would instead be written as:

6 > 4 && 6 > 8  
6 > 4 and 6 > 8

This would evaluate too false as:

true && false  
True and False

**Relational operators:**

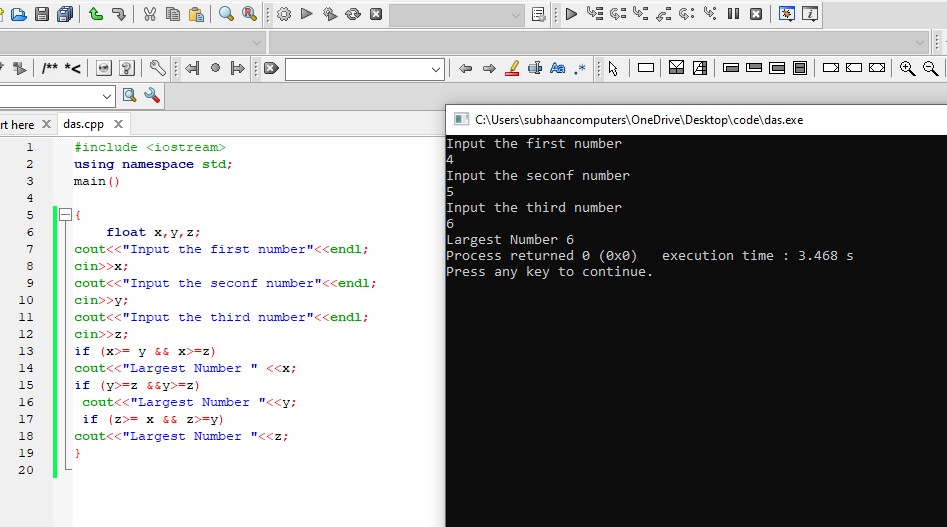
Relational operators are also known for comparison operators. Relational operators are used to relating the condition, that is it compares the two values and prints the result. In this article, we are going to see those relational operators in C++ with the help of examples.

### Different Relational Operators in C++:

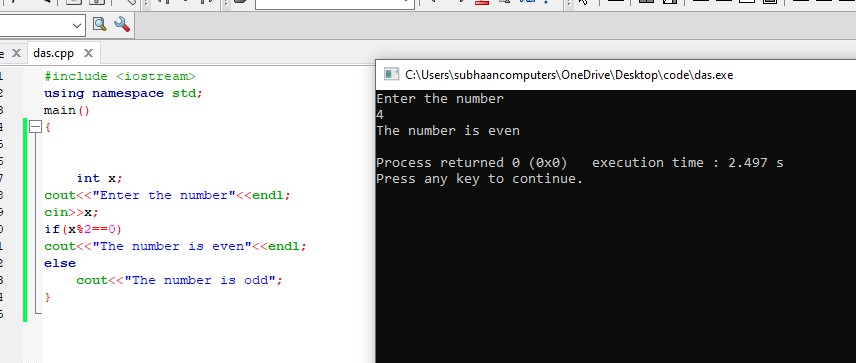
There is total 6 relational operators which are given below;

* ==
* !=
* <
* >
* <=
* >=

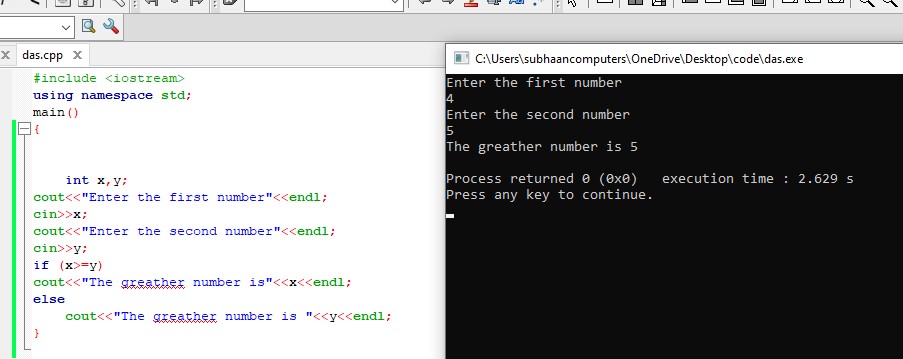
**Task NO 1**

****

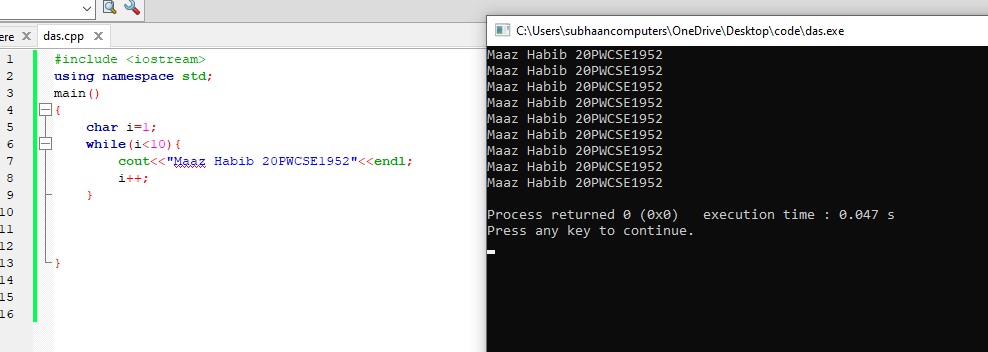
**Task NO 2**

****

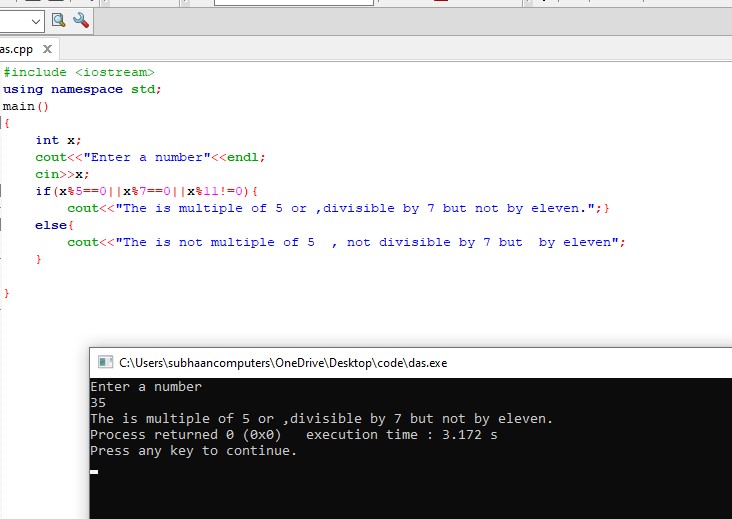
**Task NO 3**

****

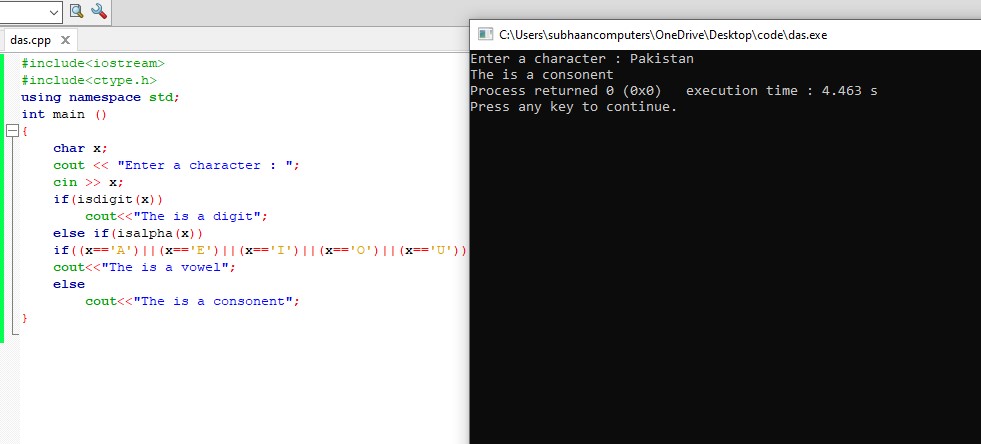
**Task NO. 4**

****

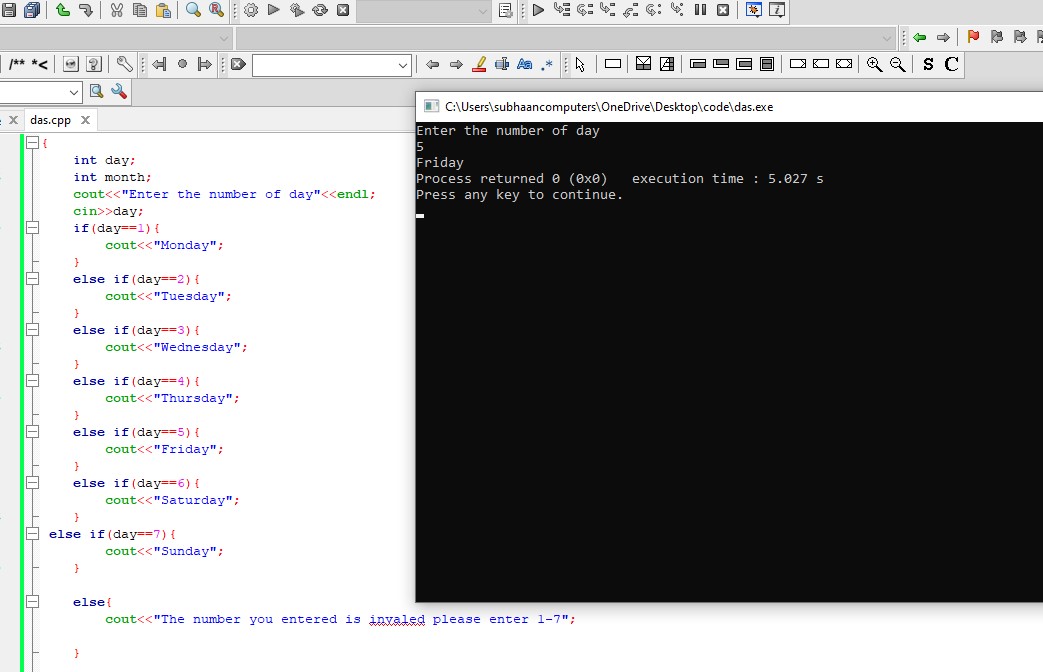
**Task NO.5**

****

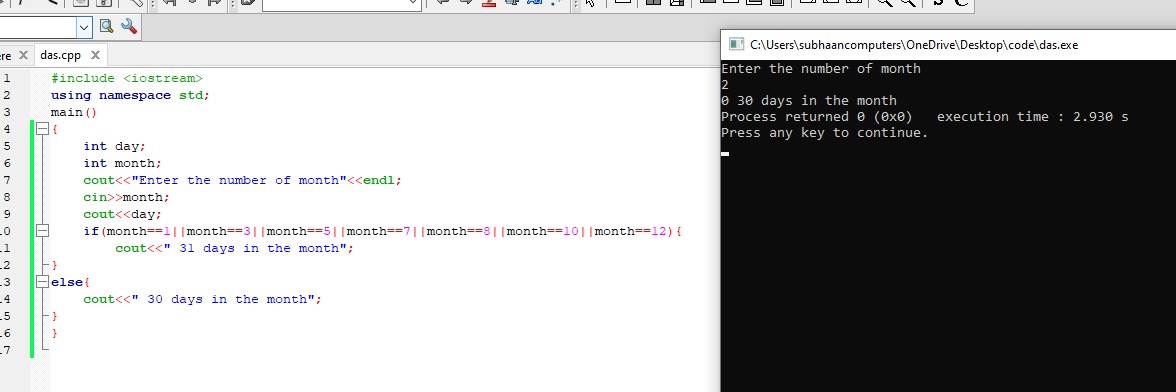
**Task NO 6**

****

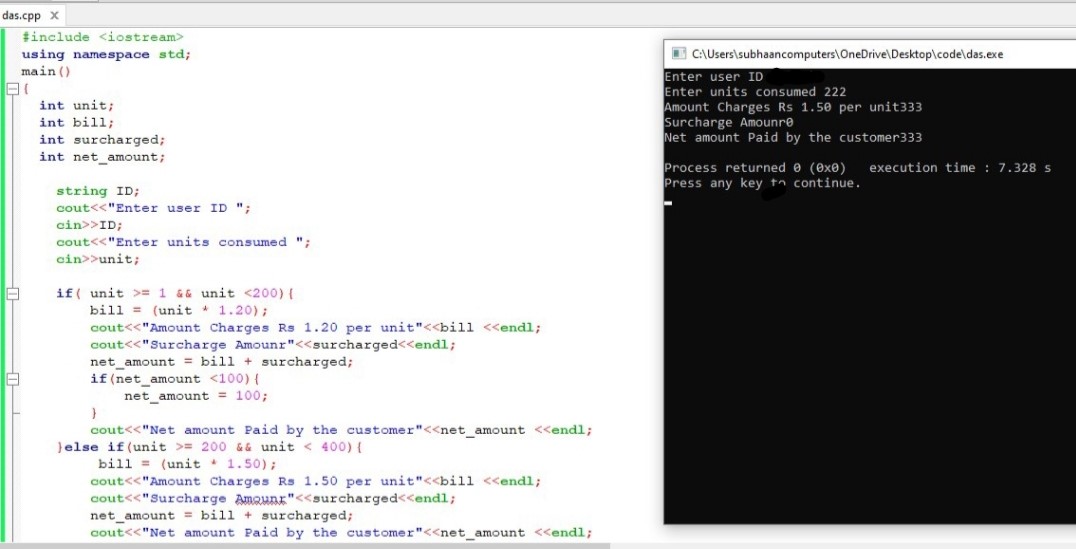
**Task NO 7**

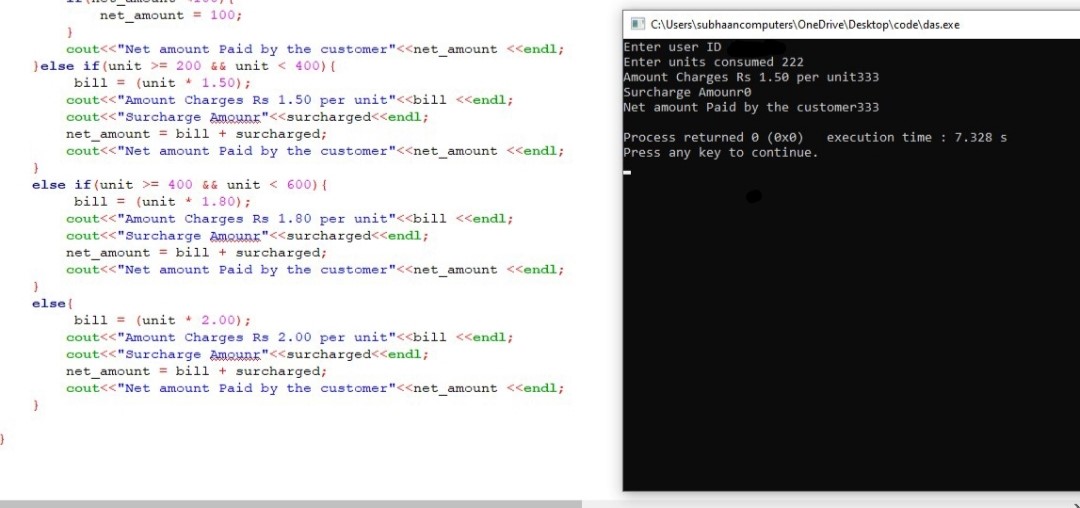
****

**Task NO 8**

****

**Task NO.9**

****

****